AFRL-HE-WP-TR-2002-0210

UNITED STATES AIR FORCE RESEARCH LABORATORY

Intelligent Mission Controller Node

David Perme

Gestalt, LLC
1040 First Avenue
Suite 302
King of Prussia, PA 19406

Matthew T. Eaton

Air Force Research Laboratory

May 2002

Final Report for the Period October 1998 to May 2002

20030513 090

Approved for public release; distribution is unlimited.

Human Effectiveness Directorate
Deployment and Sustainment Division
Sustainment Logistics Branch
2698 G Street
Wright-Patterson AFB OH 45433-7604

NOTICES

DISCLAIMER

This Technical Report is published as received and has not been edited by the Air Force Research Laboratory, Human Effectiveness Directorate.

TECHNICAL REVIEW AND APPROVAL

AFRL-HE-WP-TR-2002-0210

This report has been reviewed by the Office of Public Affairs (PA) and is releasable to the National Technical Information Service (NTIS). At NTIS, it will be available to the general public, including foreign nations.

This technical report has been reviewed and is approved for publication.

FOR THE COMMANDER

Mark M. Hoffman

MARK M. HOFFMAN
Deputy Chief
Deployment and Sustainment Division
Air Force Research Laboratory

REPORT DOCUMENTATION PAGE

Form Approved
OMB No. 0704-0188

1. AGENCY USE ONLY *(Leave blank)*

2. REPORT DATE
May 2002

3. REPORT TYPE AND DATES COVERED
Final - October 1998 - May 2002

4. TITLE AND SUBTITLE
Intelligent Mission Controller Node

5. FUNDING NUMBERS
C: F33615-99-C-6003
PE: 63231F
PR: 4923
TA: 04
WU: 02

6. AUTHOR(S)
David Perme, Matthew T. Eaton

7. PERFORMING ORGANIZATION NAME(S) AND ADDRESS(ES)
Gestalt, LLC
1040 First Avenue
Suite 302
King of Prussia, PA 19406

8. PERFORMING ORGANIZATION REPORT NUMBER

9. SPONSORING/MONITORING AGENCY NAME(S) AND ADDRESS(ES)
Air Force Research Laboratory, Human Effectiveness Directorate
Deployment and Sustainment Division
Air Force Materiel Command
Sustainment Logistics Branch
Wright-Patterson AFB OH 45433-7604

10. SPONSORING/MONITORING AGENCY REPORT NUMBER
AFRL-HE-WP-TR-2002-0210

11. SUPPLEMENTARY NOTES

12a. DISTRIBUTION AVAILABILITY STATEMENT
Approved for public release; distribution is unlimited.

12b. DISTRIBUTION CODE

13. ABSTRACT *(Maximum 200 words)*

The goal of the Intelligent Mission Controller Node (IMCN) project was to improve the process of translating mission taskings between real-world Command, Control, Communications, Computers, and Intelligence (C4I) systems and Joint Synthetic Battlespace (JSB) simulation environments. Using a rule based expert system, IMCN refines the air mission tasking from the C4I system prior to editing by mission controller personnel who are supporting the simulation exercise. This refinement includes processing the mission tasking by adding equipment loads, performing ingress and egress routing, and resolving data ambiguities. The mission controller can also perform additional refinements to the mission as well as adding new rules to further increase the fidelity of the tasking. The result is increased realism in simulation execution of planned missions and a significant reduction in the time required by man-in-the-loop controllers to process an Air Tasking Order (ATO) prior to insertion into a simulation. Equally important, the feasibility of employing expert system technologies to reason over the entire spectrum of air battlespace components was demonstrated.

14. SUBJECT TERMS
Expert System   Intelligent Mission Controller Node (IMCN)   Intelligent Agent
Joint Synthetic Battlespace (JBS)

15. NUMBER OF PAGES
20

16. PRICE CODE

17. SECURITY CLASSIFICATION OF REPORT	18. SECURITY CLASSIFICATION OF THIS PAGE	19. SECURITY CLASSIFICATION OF ABSTRACT	20. LIMITATION OF ABSTRACT
UNCLASSIFIED	UNCLASSIFIED	UNCLASSIFIED	UL

Standard Form 298 (Rev. 2-89) (EG)
Prescribed by ANSI Std. 239.18
Designed using Perform Pro, WHS/DIOR, Oct 94

THIS PAGE INTENTIONALLY LEFT BLANK

PREFACE

This research was accomplished as part of the Agent-Based Modeling and Behavioral Representation (AMBR) project, managed by the Air Force Research Laboratory, Human Effectiveness Directorate, Sustainment Logistics Branch, contract number F33615-99-C-6003. The contract was originally awarded to WPL Laboratories, which subsequently changed its name to Breakaway Solutions, Inc. Breakaway was acquired by Gestalt LLC, who then completed the effort. The period of performance for this task was from October 1998 through May 2002.

TABLE OF CONTENTS

TABLE OF FIGURES

THIS PAGE INTENTIONALLY LEFT BLANK

1. INTRODUCTION

IMCN is an expert system tool designed to aid mission controllers at Joint Synthetic Battlespace (JSB) exercises. It assists them with mid-level air mission planning between the Air Operation Center (AOC), the training audience, and the simulator.

Gestalt LLC was awarded a contract to investigate, design, and develop a prototype Intelligent Controller Node (ICN). It augments human operators serving as mission controllers during Air Force theater-level exercises. This contract comes under the Agent-Based Modeling and Behavioral Representation (AMBR) project managed by the Air Force Research Laboratory (AFRL) located at Wright Patterson AFB.

The initial contract was an 18-month effort beginning in July 1999 with the prototype demonstrated at JEFX 2000. There were three extensions to the contract to add interfaces with other simulators, a rule editor, support exercises, and to tightly integrate with the ATI system. The result of the contract and follow on phases is the IMCN system. This paper reviews the technical challenges encountered and outlines the solutions developed to address these challenges.

1.1 Project Objectives

The goals of the IMCN were:

- Reduction of cost for associated with conducting a Joint Synthetic Battlespace (JSB) exercise.
- Increase the exercise realism.
- Integrate new technology into JSB exercises.

The C4I system interface to simulators requires a technical staff. The model and mission controllers refine the command and control messages to resolve data ambiguities and differences in the way simulation and C4I systems process data. For AOC training the mission controllers spend a lot of time ensuring simulation order syntax, routing air missions, verify the mission Standard Conventional Load (SCL), select equipment, mission quality assurance, etc. The use of an expert system to aid mission controllers with mission editing will reduce exercise cost and increase exercise realism.

The focus of the IMCN project is to model the command and control echelons, and to simulate complex human behavior in order to increase the performance of the model controllers, mission controllers, and support cell operations.

1.2 Operational Environment

The Air Operation Center (AOC) is the senior element in the US Air Force's Theater Air Control System (TACS). The AOC directs the centralized functions of planning, direction, and control over deployed air resources. The primary products for the AOC to communicate air operations planning are the USMTF Air Tasking Order (ATO), Airspace Control Order (ACO) and Special Instructions (SPINS). The construction of the ATO and ACO messages require hundreds of personnel beginning 72 hours prior to the effective time. Operationally the ATO is released from the AOC with the "best effort" planning and coordination. The ATO is transmitted to the participating Wing Operation Centers (WOC) and through the Squadron Operation Centers (SOC) to the designated aircrews.

At each step, various levels of scrutiny and additional planning are applied against the actual tasking. The AOC releases an ATO to mission ready aircrews that have been trained and are well qualified to fulfill the additional planning and coordination tasks necessary for successful execution of the mission objectives. Additional planning generally takes the form of ingress and egress routing, resolution of weapon loads, resource allocation, mission element coordination, and timing adjustments. Ingress and egress routing are based upon current intelligence gathering, altitude assignments, refueling offloads, weapon range calculations and tactics.

1.3 Simulation Environment

TBMCS is used to generate the USMTF ATO and ACO messages in the operational and simulation environment. The AWSIM TBMCS Interface (ATI) processes the messages and generates Air Warfare Simulation (AWSIM) order stacks[1]. Mission controllers emulate the hundreds of operational mission planners not present at the exercise. They refine the mission planning by using the Mission Planner Workstation (MPW) to edit the order stacks. The order stacks are edited by adding the appropriate equipment, ingress and egress routing, resolution of data ambiguities, equipment standoff locations, etc. The mission controllers must finish quickly enough to allow the mission to execute and meet the timelines established by the AOC staff. The order stacks are submitted to the AWSIM simulator and the mission results are sent to the AOC staff via the ATI feedback to TBMCS.

The realism and quality of the mission controller editing is affected by several factors:

- **The ATO message length.** The number of missions in the ATO can vary from 1 to over 2000[2] depending on the objectives of the JFACC and the scale of the exercise.
- **The time available to perform mission editing.** As few as two hours to as many as twelve hours may be available for editing. A 30-member team working 150 missions for twelve hours can spend over 2 person hours per mission. For 2000 missions over two hours, less than 2 person minutes per mission is available.
- **The number of mission controllers available to support the exercise.** Most simulation centers have a staff of dedicated model and mission controllers. Military personnel or contractors may supplement this staff to support an exercise. Personnel availability can be limited by schedule conflicts, travel budgets, vacations, etc.
- **The mission controller's knowledge of air operations**. The mission controller staff normally trains supplemental controllers prior to the exercise over a two-day period. . Personnel with prior experience in day-to-day air operations are more valuable then supplemental model controllers without operations experience. Experience levels can range greatly from extremely knowledgeable to no prior experience or training. People experienced in air operations and the simulator are a rare find.
- **The level of mental fatigue of the mission controller.** Exercise schedules can run from 12 to 24 hours a day, 7 days a week. Mission controllers often work fourteen to twenty one straight days on twelve-hour shifts. A simulation center may run ten to twelve exercises or events during the year. Prolonged work schedules have a direct impact on the controllers' mental and physical capabilities. The simulation centers will attempt to stand down for 3 to 4 days after an exercise.
- **The quality of the ATO and ACO messages.** ATO and ACO messages may be syntactically correct but not executable without further work. Different AOC may use the same field to convey different meaning. Route information may be blank, define a specific path, or simply indicate use of a defined airspace. Equipment information may similarly be blank or ambiguously defined, requiring the controller to perform additional research prior to final editing. For personnel unfamiliar with air planning and operations this type of editing may be daunting. Then the mission controller must understand the limits of the simulator. For example, there are numerous mission types in the ATO message but AWSIM handles about 20 mission types. The mission controller must map the ATO mission type to something that AWSIM can handle.

There are many examples to illustrate the complexities confronting the mission control staff. Tools that alleviate redundant work or simplify mission editing will free up the mission controller to concentrate on other areas of mission planning. Using an expert system to allow senior model controllers the ability to enter rules affecting mission planning allows their expertise to be expanded across a broader range of missions. One rule can be selectively fired across all of the missions in an ATO instead of the mission controller being able to affect a selected set of missions assigned to them. The result of this tool is a reduction in the number of hours required to refine mission tasking for simulation use. More importantly, it allows a senior controller the ability to leverage his experience and knowledge across many more

---

[1] The AWSIM simulation has an API for order input. The API is textual based. An order stack refers to the mission orders generated for an ATO.

[2] These numbers are based on Gestalt LLC experience in exercises from April 1997 to present day.

missions with the expert system, rather than without it. Lastly, it offers the novice controller the ability to reuse the expertise of a trained controller to more accurately affect mission execution.

1.4 IMCN Solution

The IMCN system is an expert system aiding the mission controller in refining and expanding a mission plan. It is a sub-module of the ATI and MPW system, which handles the parsing of the ATO into C2DIF order stacks, IMCN uses the JESS inference engine to process the data received from TBMCS by ATI and fire rules when appropriate. Mission controllers are able to add, modify, or delete rules to get the desired behavior in the missions. For instance rules can be developed to decide an appropriate equipment load for a mission based upon the type of aircraft and what the mission is to accomplish. Another example, the mission controller can modify rules to specify altitude ranges for an aircraft based upon aircraft and mission type. This capability allows a mission controller to spread their expertise across all missions in the ATO. MPW, another sub-module of ATI, is used to edit the C2DIF representation of the mission order, before or after processing by IMCN.

The mission controller's workload is reduced due to IMCN. They can spend time on additional mission planning refinement that they were previously unable to perform. Over the course of time, as rules are developed, the IMCN system is "trained" to do mission editing. The mission controllers will realize an increase in the amount of time available to review and refine missions. This also allows a smaller number of mission controllers to support an exercise with the aid of the IMCN system. It frees up mission controllers to work in areas that traditionally receive less support such as threat cell augmentation or planning on complicated mission tasking. As the IMCN knowledge base grows, the ability of the system to process missions outperforms an inexperienced mission controller. The IMCN system is never mentally fatigued and will apply a steady set of logic across all missions throughout the duration of the exercise.

2. PROJECT

2.1 Phase 1

The first phase began July 1999 for 18-month to conclude with a demonstration at an Air Force exercise. Gestalt LLC put together a team possessing a great deal of experience in the JSB environment. This team was rounded out with members experienced in artificial intelligence and expert systems.

The National Air and Space Warfare Model (NASM) was considered but it is not ready to be used for the proof of concept test. The AWSIM simulator was chosen since it is used at the JSB simulation centers, has an experienced mission controller staff, and the simulation centers are looking to improve exercise realism and cost reduction.

AFRL requested that IMCN be developed as a package that can be integrated with existing systems used by the simulation centers. This required IMCN to work within the JSB environment and not interfere with the exercise during the proof-of-concept demonstration.

The design required an interface to the TBMCS Command and Control system, an interface to the AWSIM simulator, and the ability to edit the mission orders. It was recommended in the proposal to use HLA but the JSB simulation environment does not use HLA operationally. The only system with a direct link between TBMCS and the ASWIM simulator is the ATI system.

IMCN needed a common data transport layer to communicate the mission planning and allow simulators the ability to build orders. The project began with the analysis of the USMTF ACO and ATO messages and the AWSIM, RESA, JTLS, NASM, ENWGS, and the JWARS order syntax. This effort identified the common data elements across the simulation systems and the USMTF messages resulting in the Command and Control Data Interchange Format (C2DIF) data structures.

IMCN uses the ATI, MPW, and GIAC software that the JSB simulation community is familiar with.

Several mission controllers and ATI operators were contacted to compile a list of tasks that would reduce their workload. The top two responses were ingress and egress routing to handle threat avoidance and loading of equipment based on aircraft and mission type. They also requested package planning rules, refuel planning, the ability to use the SPINS information, and the ability to resolve ATO data ambiguities. The team focused on solving the first two issues and working with mission controllers to add further IMCN capabilities over time.

2.2 Phase 2

This was a small phase and the goal is for ATI / IMCN to interface with the RESA simulator and demonstrate the capability at a JTASC exercise. The RESA and AWSIM simulator branched from the same naval simulator, and therefore, are very similar in order syntax.

RESA was one of the simulators evaluated when C2DIF was developed. IMCN internally was already able to handle the RESA data requirements. The bulk of the work was interfacing ATI with RESA to pull data from the simulator and send data to the simulator. Another large effort involved getting the RESA simulator to running at the test laboratory with an unclassified data source. The system was demonstrated at the JTASC for the Internal Look exercise.

2.3 Phase 3

The phase 3 effort called for developing a rule editor and to interface ATI / IMCN with the SOAR simulator. The demonstration for the SOAR interface happened at FBE-I.

Crafting rules for a knowledge base is a daunting task for model controllers that do not have expert system experience. The typical model controller will be technically sound as a system administrator, some programming experience, and trained on the simulator that they operate. It was recognized that a user-friendly interface is needed for the model controllers to craft rules.

The interface to the SOAR simulator is based upon XML messages. The C2DIF from ATI/IMCN generates route, SCL, and air mission messages. The SOAR simulator executes the air mission and submits the mission result to ATI. The ATI send the result message to TBMCS for mission feedback to the training audience.

2.4 Phase 4

The goal for phase 4 is to prepare IMCN for acceptance into the suite of simulation tools. The integration between ATI and IMCN will become tighter and robust for the operational environment. The system will support exercises or demonstrations in Korea, Germany, and C2TIG to show its capabilities to the simulation audience. An interface to IMCN from NASM will be developed to allow ingress and egress routing for air missions.

The Ulchi Focus Lens (UFL) 2001 exercise trains the US Joint Forces and the Republic of Korea (ROK) forces. The focus for IMCN was base logistics. IMCN would compare the number of munitions being loaded on the aircraft with the weapon stores on the base.

The focus for RSOI 2002 is a logistics exercise for the US Air Force and the ROK forces. IMCN provided fuel warning and threat voidance for the missions.

AFAMS requested a testing of the IMCN software at their facility. They had an Ultra 60 for the AWSIM simulator to run on. A SUN Sparc 10 was used for the Oracle database server. The ATI software and the IMCN software were going to run on a SUN Ultra 2. The operating system was not patched with the SUN OS 2.7 recommended patches. The testers wanted to use the ATO messages from UFL 2001. During UFL the simulation center used SUN Ultra 60 systems to support the ATI and IMCN software. By the time the patches were installed and the ATO's were processed there was a half a day to examine the order stacks.

A demonstration of the IMCN system was held at the C2TIG facility. ATO messages were processed without and with the IMCN input to the messages. The messages were displayed side by side. The controllers were brought down to review them and make recommendations. During the evening, the IMCN developers wrote rules to set the takeoff speed and attitudes for helicopters, A10 aircraft, and aircraft type. The next day the controllers were able to see how the new rules and facts affected the mission orders. They most common asked question by the controllers is, "When do we get the system?"

IMCN was used for Enable Freedom 2002 at the Warrior Preparation Center (WPC). It fixed simulation anomalies, mission routing, weapon selection, fuel warnings, and take off attributes. It processed 2245 missions and made over 7000 changes to the orders.

The IMCN system has received letters of recommendation from AFAMS, C2TIG, and WPC. They see the IMCN system adding value to the simulation community and desire continued funding to the system.

3. IMCN COMPONENTS

The IMCN software will run on any operating system supporting the JAVA language. It is developed with JAVA 1.3.1 and it is being tested with JAVA 1.4. The simulation centers normally use the Solaris 2.7 or 2.8 on SUN Ultra machines. The Solaris 2.7 recommended patches must be installed to support JAVA 1.3.1.

At UFL 2001 and RSO&I 2002 a SUN Ultra 60 with 1024 megabytes of RAM was used. At Enable Freedom 2002 a SUN Ultra 30 with 1024 megabytes of RAM was used. For the FBE-I and FBE-J exercises the IMCN software is running on Linux.

The ATI software must be run on a Solaris system in order to interface with the TBMCS system. The ORACLE database is also required for ATI.

3.1 AWSIM TBMCS Interface (ATI)

IMCN needed an interface to the C4I system and the simulator. The ATI (and its predecessor system, the AWSIM CTAPS Interface (ACI)), system has been used by the JSB communities since 1996. The system is based on the initial design of Project Real Warrior (PRW) developed at the Warrior Preparation Center (WPC) in 1995. PRW was one of the initial prototypes to interface operational C4I systems, in this case CTAPS, with the JSB simulators. The PRW prototype evolved into ACI and finally, with the AOC shift from CTAPS to TBMCS, into what is the current ATI system.

The ATI system has five main functions:

- Extract Scenario data from the TBMCS Air Operations Database (AODB).
- Map simulation database entries to the TBMCS database entries.
- Translate the ATO and ACO messages.
- Provide a Mission Planner Workstation (MPW) for elaboration, checking and editing of the ATO missions.
- Provide mission feedback to TBMCS.

Figure 1 shows the ATI system architecture which details the data flow between TBMCS, AWSIM and IMCN. The ATI system receives Airspace Control Order (ACO) and Air Tasking Order (ATO) messages from the TBMCS system. These messages are parsed into an internal database representation for easy data quality editing of missions. The ATI operator edits lookup tables with data needed by AWSIM to execute orders. The GIAC Data Server (GDS) interfaces with AWSIM to pull information from the simulator that ATI will need for data quality checks between TBMCS and AWSIM. The model controller also uses the Graphical Input Aggregate Control (GIAC) to view the mission.

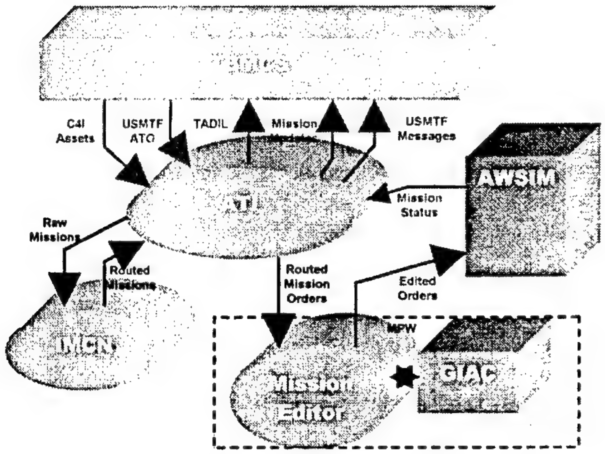

Figure 1: ATI System Architecture

The Mission Planner Workstation (MPW) gives the model controller the ability to view the ATO, view and modify AWSIM order stacks, and visually modify missions using GIAC.

By interfacing IMCN with ATI, existing interfaces to TBMCS and the simulators are reused. IMCN utilizes the capabilities of the ATI system providing model controllers a familiar system in which to perform mission editing without the need to learn an entirely new environment.

3.2 Java Expert System Shell (JESS)

IMCN uses the Java Expert System Shell (JESS) to perform reasoning on raw C4I data. JESS is a rule engine and scripting environment written entirely in Sun's Java™ language by Ernest Friedman-Hill at Sandia National Laboratories. Jess was originally inspired by NASA's CLIPS expert system shell, but has grown into a complete, distinct, dynamic environment of its own. Using Jess, you can build Java applets and applications that have the capacity to "reason" using knowledge you supply in the form of declarative rules. JESS is open source software,
See http://herzberg.ca.sandia.gov/jess/

IMCN is itself a pure Java application, making JESS an ideal candidate for an expert system layer. IMCN uses C2DIF mission data, in the form of Java objects. C2DIF mission objects can be bound to JESS variables or inserted into JESS facts. JESS uses Java reflection to provide a full suite of specialized functions to create, query and manipulate arbitrary Java objects without prior knowledge of their structure or contents. Java objects may even be asserted directly into the JESS expert system environment using the built-in JESS functions 'Defclass' and 'Definstance'. The resulting JESS 'shadow facts' closely mirror the structure of the original C2DIF objects, and use Java's native event messaging system to synchronize JESS facts with the original Java object containers without the need of additional coding.

Like CLIPS, Jess uses the Rete algorithm to process rules -- a very efficient mechanism for solving the difficult many-to-many matching problem. Jess is also a powerful Java scripting environment, from which you can create Java objects and call Java methods without compiling any Java code.

Whenever a rule fires and alters a set of facts, all of the Rules must be evaluated again to see if they now apply. Rete makes JESS much faster than a set of cascading if.. then statements in a loop. An expert system shell such as JESS allows the user to define a simple set of mutually interacting rules with which to model extremely complex behavior in a fast and efficient manner. The user may even alter facts and rules 'on the fly' to examine their effects without requiring code to be recompiled, or even for the system to be shut down or reset. This is especially useful in a dynamic and intensely human interactive environment such as occurs in a simulation exercise.

3.3 HLA

Phase I for the IMCN system required a HLA interface; however, none of the simulators in the JSB environment were using HLA. An interface with the RTI software was developed and used in the laboratory for a proof-of-concept. A C2DIF FOM was used to pass objects out and back into IMCN.

For Phase 4 the IMCN system is being interfaced with the NASM simulator. The NASM system is going to use the IMCN route planner to avoid threats for mission planning. The RTI interface is updated to work with the JSIMS FOM. The most challenging piece is mapping the JSIMS FOM with the IMCN C2DIF classes. It is being tested at the writing of this document.

3.4 IMCN Software Modules

IMCN was developed as stand alone software modules. This allowed the modules to be developed and tested in parallel development. The modules were then joined to a message hub for communication. Each module is an independent thread and the message interactions (Figure 2) between the modules are event driven. The IMCN GUI gives the operator control over each module.

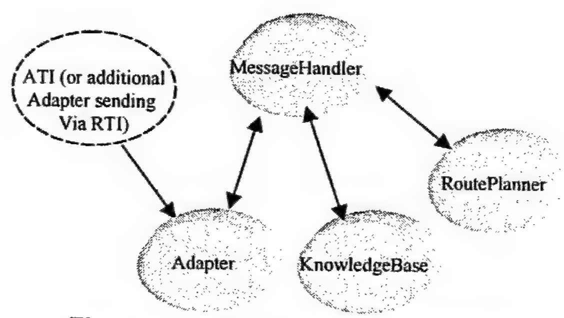

Figure 2: IMCN System Interactions

3.4.1 C2DIF

IMCN needed to operate with the group of simulators used by the JSB simulation centers. There was a need for a normalized data layer that the simulators can use to build their orders from. The Command and Control Data Interchange Format (C2DIF) is a package of classes that provide the normalized data layer capturing the planning of the ATO and contains the data that simulators require to generate orders.

Identifying the data required for each simulation order and the USMTF ATO and ACO messages drove the development of the IMCN C2DIF classes. A spreadsheet of the data fields for AWSIM, RESA, JTLS, and NASM simulators and the USMTF ACO and ATO messages. Each document was merged and refined into a master document and the result is the IMCN C2DIF.

The airspace C2DIF communicates the air space defined by the ACO messages. The air mission contains a list of events that the mission will execute based upon sequence, time, or location. For example the event list would be takes off, proceed to air space, refuel at some time, and land at some time. An order writer exists for each of the different simulators that generate an order stack from the C2DIF air mission object. Figure 3 shows an example of the C2DIF classes.

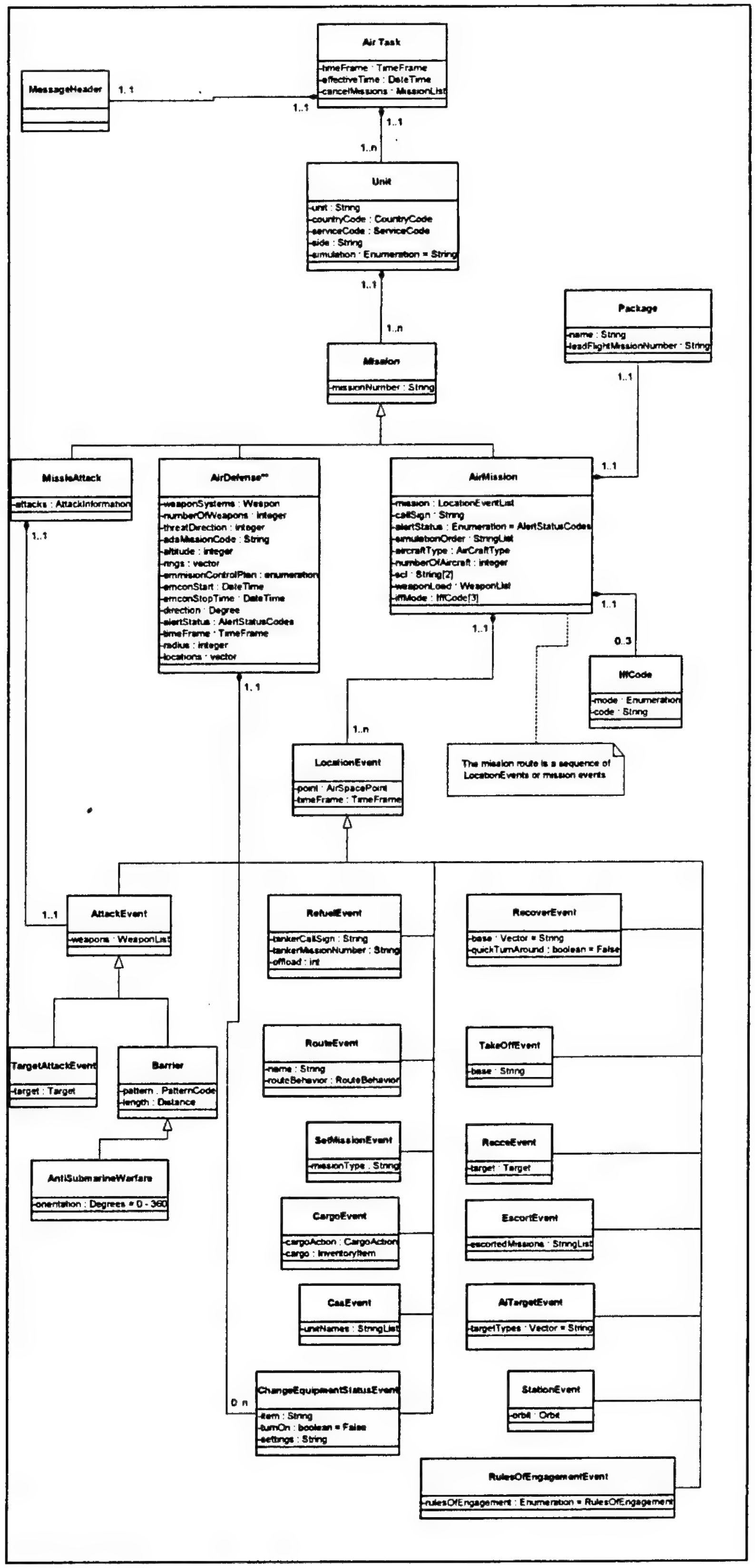

Figure 3: Example of C2DIF Classes

3.4.2 *IMCN Adapter*

The adapter is a general-purpose interface for clients to connect to IMCN. Each client has a software piece that converts from its internal data structures to CDIF and back again. This software is layered on top of the adapter. It is currently used to interface with ATI through JDBC, with NASM through RTI, and with JSAF through XML messages.

3.4.3 *Message Handler*

The message handler is the hub of IMCN that all modules attach to. Each module attaches to the message handler and identifies the message types that they use. When a message is received it is sent forward to all modules registered for the message type.

3.4.4 *Knowledge Base (KB) Manager*

The knowledge base manager is built around the JESS engine. The objects received by the KB manager are mirrored as "shadow" facts within the JESS knowledge base. A special 'OBJECT' slot is used as a unique identifier, similar to the 'primary key' in a database table. The 'OBJECT' slot also provides a handle on the original Java object, allowing the methods to be called. The full membership hierarchy of the original Java objects is captured in the 'OBJECT' cross references, while Class inheritance is captured using the JESS 'extends' modifier in the fact templates.

IMCN builds a knowledge base from a set of aircraft, weapon and mission properties derived from the simulation, and a set of pre-defined global and local rulesets. Missions in the form of C2DIF objects are then added to the knowledge base for analysis and modification. Rules fire as matches on facts are discovered. The knowledge base may send the mission to other analysis modules such as the route planner, select an appropriate equipment load based upon the mission and target parameters, or alter the mission based upon other rules that have been entered on the fly. A rule editor has been added to IMCN that provides a visual interface for rule management by the model controller. The model controller is able to enable or disable various rule-sets interactively, or modify them 'on the fly' using a simple graphical interface. The more experienced model controller also has a direct command-line interface to the knowledge base.

IMCN gives model controllers the ability to add, delete, or modify rules that can act upon the C2DIF classes. The expert system applies the model controller rules and facts to refine the mission planning. This will enable the expert system to perform the bulk of the mission routing, weapons selection, aircraft type selection, attack target location, etc., leaving the model controller to review the missions and focus on the more difficult cases. If the model controller observes a mission behavior that is not valid, then a rule can be added or modified to correct the errant behavior. This rule can then be applied to all subsequent missions and possibly stored for permanent inclusion into the local or global knowledge bases.

Here is an example of a rule to set the takeoff speed of an aircraft in a mission to the cruise speed as defined in the aircraft property fact (comments begin with a semicolon):

"Set all of the takeoff speeds to the slot take off speed".
(defrule takeOffSpeedRule
 (declare (salience 10)) ;set rule priority.

This is the 'if' side of the rule.

```
 ?f <-
  (AirMission
   (missionObject ?airMissionObject)
   (missionNumber ?number)
   (aircraftType ?aircraftType)
   (takeOffSpeed 0.0)
   (eventObjectArray $? ?TakeOffEventObject
$?)
```

```
  )
    (aircraft
      (name ?aircraftType)
      (cruiseSpeed ?speed)
  )
  =>
```
This is the 'then' side of the rule

```
 (if    (instanceof     ?TakeOffEventObject
 TakeOffEvent)
 then
   (call                  ?TakeOffEventObject
 setSpeedKilometers ?speed)
   (printout t "Mission " ?number " take off speed
 " ?setSpeed crlf)
   )
 )
```

3.4.5 Route Planner

The route planner (Figure 4) tracks the threats and the air spaces. When it receives an air mission it selects the route with the least threat from point A to B. The missions in the ATO do not have threat avoidance. The AOC planners are not concerned with ingress or egress routing for threats. It is up to the flight commander since they will have the current threat information. During an exercise this falls upon the mission controllers.

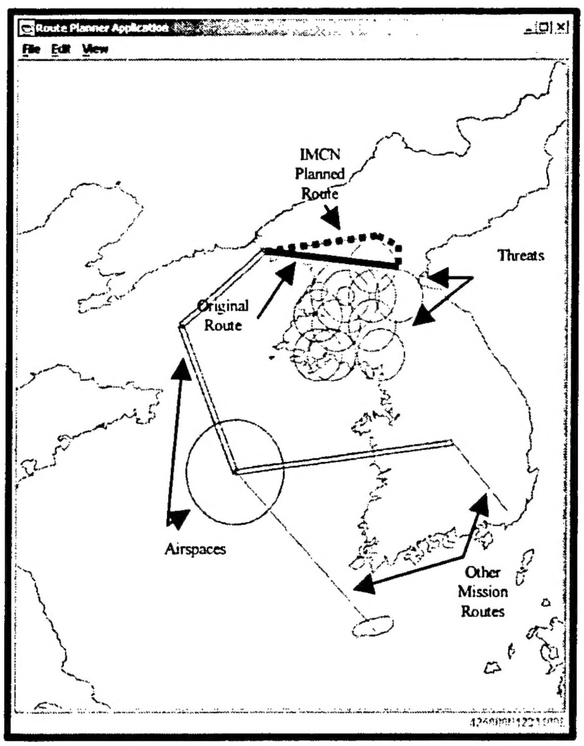

Figure 4: Route Planner

With the introduction of the Common Operational Picture (COP) and the Situational Awareness and Assessment (SAA) system, the AOC can now monitor graphically the flightpath of all tasked missions. Therefore, the addition of realistic routing is a benefit in two ways; (1) Adds to the realism of the exercise experience for the NAF and (2) allows the simulation to use real world performance attributes, such as

Probability of Kill for weapons parameters. For much of the history in computer aided exercise events (CAX), the lack of the ability to have realistic routing, both rear area and threat avoidance led to unacceptably high and artificial "kill rates". These rates were and are mitigated by "tuning" such parameters as Probability of Kill and Probability of Launch to account for the erroneous overflight of SAMs by missions. Realistic routing consistently applied to all missions removes this artificiality and adds to the credibility of the exercise.

The route planner is responsible for estimating ingress and egress routes for the mission around or through the known threat areas. The mission controller can decide to keep the mission routing or modify the route path through the MPW software. The routing is accomplished by examining the battle space grid that contains the known threat areas and defined air spaces for the battle space. The route planner will try to avoid threat areas based upon the altitude that the aircraft is flying and the fuel constraints of the aircraft. The route planner does not take the mission time constraints into consideration when choosing a path at this time; however, this capability can be added. Route planner will try to use defined air spaces identified as desired and avoid the defined airspaces identified as restricted. If the mission contains a defined airspace or location to use then route planner will not modify the defined air space. It will plan a route to the defined airspace and away from the defined airspace. Anything within the defined airspace will not be changed. The distance of the defined airspace will impact the route planning distance calculations.

3.4.6 Rule Editor

The rule editor (Figure 5) assists the model controller with adding, modifying, or removing facts and rules from the inference engine. The C2DIF hierarchy is displayed in a tab pane, while the class hierarchy, and fact templates are displayed a pane. By selecting nodes on the C2DIF or KB panes the corresponding fact patterns are generated and added to the rule editor window. The C2DIF and KB panes may also be searched for keywords. Matching nodes in the search result table displays and selects the corresponding node.

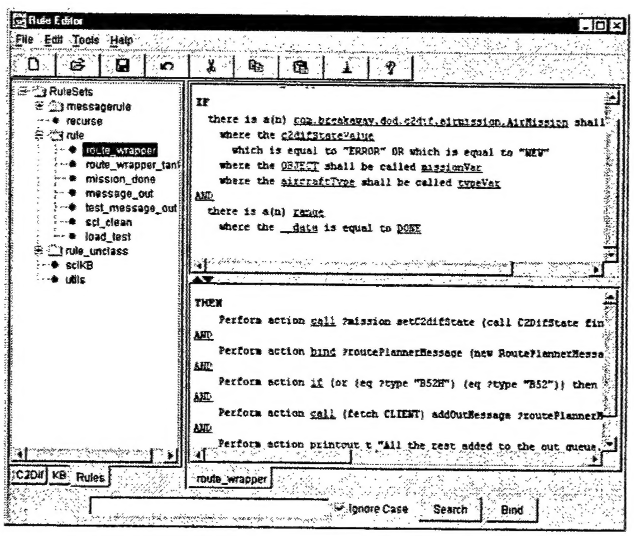

Figure 5: Rule Editor

The model controller is an expert with the simulator and assisting the mission controllers. They are not experts in the JESS syntax or inference engines, nor are they intimately familiar with the structure of the C2DIF classes. The rule editor enables the model controller to visually write facts and rules and get the expertise into IMCN.

It became evident that those familiar with expert systems like the rule editor. But the model controller does not feel comfortable with the rule editor. To overcome this in the near term, IMCN has user interfaces designed for specific rules. The controllers are comfortable with these screens since there are detailed and very specific in scope. In time the model controllers will become comfortable with the knowledge base manager and use the rule editor

4. SUMMARY

The objectives of the Intelligent Mission Controller Node (IMCN) project were to reduce the cost of conducting Joint Synthetic Battlespace (JSB) exercises, increase exercise realism, and integrate new technology into JSB exercises.

IMCN uses expert system technologies to reduce the mission controllers' day-to-day workload during JSB exercises by assisting with mission planning. Using a rule based engine, IMCN refines the air mission tasking from the C4I system prior to editing by mission controllers. This refinement includes processing the mission tasking by adding equipment loads, performing ingress and egress routing, and resolving data ambiguities. The mission controller performs further refinements to the mission as well as adding new rules to IMCN to increase the fidelity of the tasking. The result is increased realism in simulation execution of planned missions and a significant reduction in the time required by man-in-the-loop controllers to process an Air Tasking Order (ATO) prior to insertion into a simulation.

The IMCN software was demonstrated at eight major simulation exercises supporting such organizations as the Air Force Agency for Modeling and Simulation (AFAMS), the Air Force Command and Control Training and Innovation Group (AFC2TIG), and the Warrior Preparation Center (WPC). During Enable Freedom 2002, IMCN processed 2284 missions and made 6982 changes, adding realism to the exercise and allowing the mission controllers to focus on other tasks to support the exercise. IMCN technology has been transitioned to the Electronic Systems Center and is being integrated into the Air Force Modeling and Simulation Training Toolkit (AFMSTT).

5. REFERENCES

Java Expert System Shell (JESS), Web Site http://herzberg.ca.sandia.gov/jess/

C Language Integrated Production System (CLIPS), Web Site http://www.ghg.net/clips/CLIPS.html

Grady Booch & Ivar Jacobson & James Rumbaugh (1999). *The Unified Modeling Language Reference Manual, Addison*-Wesley

Judith Dahmann & Frederick Kuhl & Richard Weatherly (1999). *Creating Computer Simulation Systems: An Introduction to the High Level Architecture,* Prentice Hall PTR

Joseph Giarrantano & Gary Riley (1998). *Expert Systems Principles and Programming: third edition,* PWS Publishing Company

Peter Jackson (1998). *Introduction to Expert Systems: third edition,* Addison-Wesley

National Research Council (1998). *Modeling Human and Organizational Behavior,* National Academy Press

Putnam Texel & Charles Williams (1997). *Use Cases Combined with Booch OMT UML,* Prentice Hall PTR

Mark Watson (1997). *Intelligent Java Applications For The Internet and Intranets,* Morgan Kaufmann Publishers